VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018

A Database Management System Mini Project Report on

"Stock Market Management System"

Submitted in Partial fulfillment of the Requirements for the V Semester of the Degree of

Bachelor of Engineering in

Computer Science & Engineering

By

MEHUL SINGH (1CR20CS114)

NEERAJ KUMAR (1CR20CS125)

Under the Guidance of,

Prof. Kartheek G.C.R, Assistant Professor, Dept. of CSE

Prof. Anjali Gupta, Assistant Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

Affiliated to VTU, Approved by AICTE, Accredited by NBA and NAAC with "A++" Grade

ITPL MAIN ROAD, BROOKFIELD, BENGALURU-560037, KARNATAKA, INDIA

CMR INSTITUTE OF TECHNOLOGY

Affiliated to VTU, Approved by AICTE, Accredited by NBA and NAAC with "A++" Grade

ITPL MAIN ROAD, BROOKFIELD, BENGALURU-560037, KARNATAKA, INDIA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Database Management System Project work entitled **"STOCK MARKET MANAGEMENT SYSTEM"** has been carried out by **Mehul Singh, 1CR20CS114 and Neeraj kumar, 1CR20CS125,** bonafide students of CMR Institute of Technology, Bengaluru in partial fulfillment for the award of the Degree of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year **2022-2023**. It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the report deposited in the departmental library. This Database Management System Project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said Degree.

Signature of Guide	**Signature of HOD**
Prof. Kartheek G.C.R	**Dr. Shreekanth M Prabhu**
Asst. Professor	**Professor & HoD**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**

External Viva

Name of the Examiners — Signature with date

1.

2.

DECLARATION

We, the students of V semester of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the project work entitled "**STOCK MARKET MANAGEMENT SYSTEM**" has been successfully completed under the guidance of Asst. Prof. Kartheek and Asst. Prof. Anjali Gupta, Dept. of Computer Science and Engineering, CMR Institute of technology, Bengaluru. This project work is submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2022-2023. The matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bangalore

Date:17-01-2023

Team members:

Mehul Singh (1CR20CS114)	
Neeraj Kumar (1CR20CS125)	

ABSTRACT

This project is aimed at developing a Web-based application named Stock Market Management System for managing the stock system of any organization. The Stock Market Management System refers to the system and processes to manage the stock with the involvement of Technology system.

Stock Market management system refers to keeping track of the ownership of the shares of a company by various users. A share represents part of a company. The Administrator maintains a record details of each shareholder, including the number of shares he owns and filing of new companies in market and updating the recent current exchange value. Stock Market Management System is an application which manages the shares of a company. It is supposed to execute common functions of stock market. A user should be able to use his own username and password to access the system and perform corresponding operations

It is supposed to execute common functions of stock market. A user should be able to use his own username and password to access the system and perform corresponding operations

ACKNOWLEDGEMENT

We take this opportunity to express my sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing me a platform to pursue my studies and carry out the Database Management System Project.

It gives me an immense pleasure to express my deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bengaluru, for his constant encouragement.

we would like to extend my sincere gratitude to **Dr. Shreekanth M Prabhu,** HOD, Department of Computer Science and Engineering, CMRIT, Bengaluru, who has been a constant support and encouragement throughout the course of this project.

We would like to thank my guide **Asst. Prof. Kartheek and Asst. Prof. Anjali Gupta,** Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of the project work.

We would also like to thank all the faculty members of Department of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, we thank our parents and friends for all the moral support they have given me during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The final project of this course this semester is the Stock market management system. It is supposed to execute common functions of stock market. A user should be able to use his own username and password to access the system and perform corresponding operations. It took us great effort to develop this application using HTML, CSS as Front end and NodeJS as Back-end interface. This software is duly tested on different environment of hardware and software. The data security of this software is great as the data stored in various tables with unreadable characters.

This application provides various data such as Stock market Business, Stock information, the Balance details of the firm, Client personal Information etc. It also provides data such as various Companies along with its Shares with specific code.

Stock Market Management System is an application which manages the shares of a company. Every organization tries to build up their resources and increase their business. For increasing the business, they need lots of funds. Raising the funds are of so many types among which issuing of shares is one of the methods. After issuing the shares a private sector converts to a public sector. When an organization becomes a public sector the responsibility and accountability also increase. The starting price at which company provides stocks to the customers is known as Initial Public Offerings (IPO's). Price of the company varies as per various conditions like inflation, recession, crisis etc. and according to price change determines the profit and loss of the customers. In this application users can also raise their concerns with the administrator.

CHAPTER 2

SYSTEM REQUIREMENTS

Firstly, allowing the end user to interact with the system through interface and provide much more enjoyable user experience, particularly for non-technical users. This website is designed in such a way that it is portable in all the system. It is designed so that it is very flexible.

2.1 Hardware Requirements

- Processor: Intel or AMD
- RAM: 1.00GB
- Monitor: Any Colour Monitor
- Processor Speed: 1.00GB

2.2 Software Requirements

- Operating System: Windows/Linux or MacOS
- Frontend: HTML, CSS, JavaScript, Bootstrap
- Backend: NodeJS, MySQL, ExpressJS
- Database Connectivity: phpMyAdmin
- Server: XAMPP

CHAPTER 3

DESIGN

It is designed using frontend Languages such as HTML, CSS, JavaScript and Bootstrap, backend is designed using NodeJS, ExpressJS and MySQL, connected using phpMyAdmin and Xampp server.

3.1 Schema Diagram Description

A schema diagram is a diagram which contains entities and the attributes that will define that schema. A schema diagram only shows us the database design. It does not show the actual data of the database. Schema can be a single table or it can have more than one table which is related.

There are total six table in the schema of this database in which holding and support table has a foreign key attribute which references to an attributes of user table names id and holding table has a foreign key attribute which references to an attribute of company table named c_id.

3.2 ER Diagram Description

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation that depicts relationships among people, objects, places, concepts or events within an information technology (IT) system.

There is total five entities in the ER Diagram in which user can be managed by admin. Whenever user needs support then it can be resolved by admin. Admin can add, delete and update the company details. A user can buy the shares of a company.

3.3 Schema Diagram

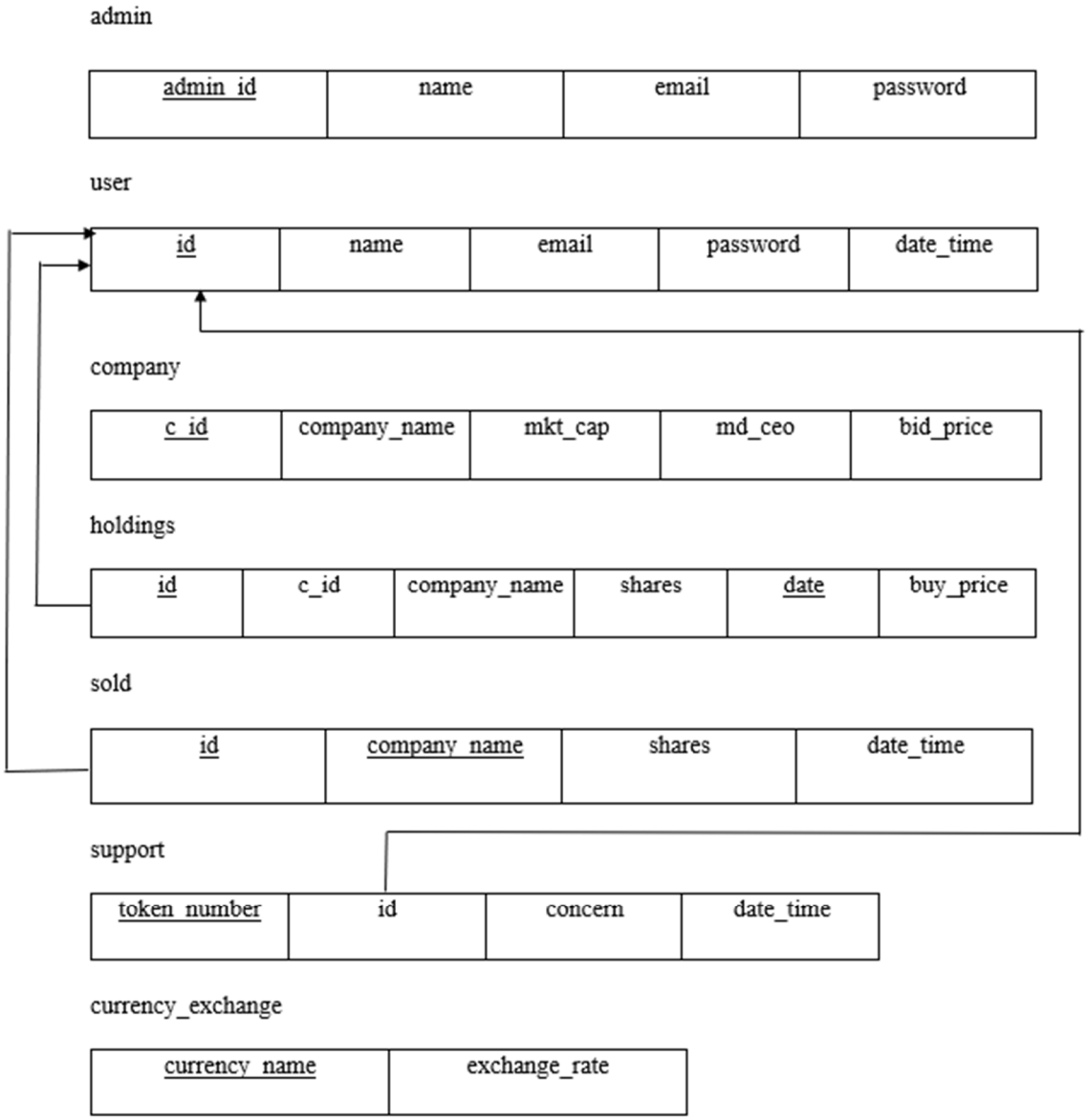

Fig 3.3- Schema Diagram

3.4 ER Diagram

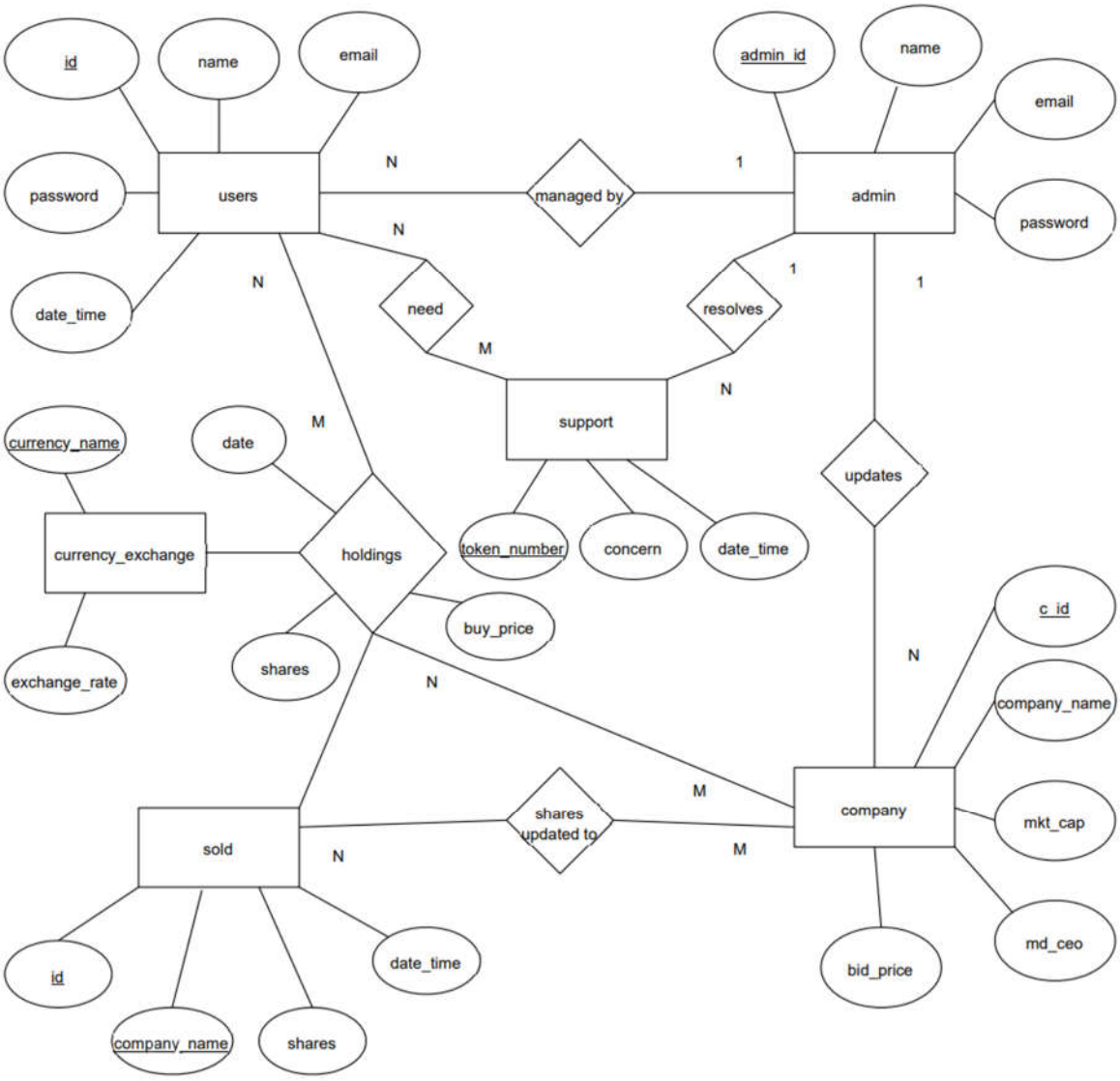

Fig 3.4- ER Diagram

CHAPTER 4

IMPLEMENTATION

The implementation of this project includes the use of NodeJS and MYSQL. This code allows to maintain the record of all users and the stocks he/she purchased. It allows an admin to add/delete company and user details. The company details can also be updated by admin and that change can also reflected for each user in his login.

4.1 SOURCE CODE:

FRONT END:

User Registration Page:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65"
crossorigin="anonymous">
   <link rel="stylesheet" href="/style.css">

   <title>Document</title>
</head>
<body>

   <nav class="navbar navbar-default">
      <h1> Registration</h1>
    <ul>
      <li><a href="/">Home</a></li>
      <li><a href="/login">Login</a></li>
      <li><a href="/register">Register</a></li>
    </ul>

   </nav>
   <div class="container mt-4">

         <h1 class='display-4'>start your investment journey by registering </h1>
         <div class="card">
            {{!-- <img class="card-img-top" src="..." alt="Card image cap"> --}}
            <div class="card-header">Registration Form</div>
            <div class="card-body">
              <form action="/auth/register" method="POST">
```

```
                <div class="form-group">
                    <label for="name">Name</label>
                    <input type="text" class="form-control" id="name" name="name"
placeholder="Enter name">

                </div>
                <div class="form-group">
                    <label for="email">E-mail address</label>
                    <input type="email" class="form-control" id="email" name="email"
placeholder="enter email">
                    <small id="emailHelp" class="form-text text-muted">We'll never share your
email with anyone else.</small>
                </div>

                <div class="form-group">
                    <label for="password">Password</label>
                    <input type="password" class="form-control" id="password" name="password"
placeholder="enter Password">

                </div>
                <div class="form-group">
                    <label for="passwordconfirm">Confirm Password</label>
                    <input type="password" class="form-control" id="passwordconfirm"
name="passwordconfirm" placeholder="enter Password">

                </div>
                <button type="submit" class="btn btn-primary">Register User</button>
                </form>
            </div>
        </div>
        {{#if message}}
        <h4 class="alert alert-danger mt-4">{{message}}</h4>
        {{/if}}
    </div>

    <script src="C:\bootstrap-5.2.3-dist\js\bootstrap.min.js">
</body>
</html>
```

4.2 BACKEND CODE (NodeJS & MySQL):

Libraries (JSON File):

```
{
  "name": "dbms",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon app.js"
```

```
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "async": "^3.2.4",
    "bcryptjs": "^2.4.3",
    "body-parser": "^1.20.1",
    "cookie-parser": "^1.4.6",
    "dotenv": "^16.0.3",
    "ejc": "^1.0.4",
    "ejs": "^3.1.8",
    "express": "^4.18.2",
    "express-session": "^1.17.3",
    "hbs": "^4.2.0",
    "jsonwebtoken": "^9.0.0",
    "mysql": "^2.18.1",
    "nodemon": "^2.0.20",
    "pug": "^3.0.2"
  },
  "description": ""
}
```

Authorization Control:

```
const mysql = require("mysql");
const jwt = require('jsonwebtoken');
const bcrypt = require('bcryptjs');
const { promisify } = require('util');

const db = mysql.createConnection({
  host: process.env.DATABASE_HOST,
  user: process.env.DATABASE_USER,
  password: process.env.DATABASE_PASSWORD,
  database: process.env.DATABASE
});

exports.login = async (req, res) => {
  try {
    const {email,password } = req.body;

    if( !email || !password ) {
      return res.status(400).render('login', {
        message: 'Please provide an email and password'
      })
    }
```

```
    db.query('SELECT * FROM users WHERE email = ?', [email], async (error, results) => {
      console.log(results);
      if( !results || !(await bcrypt.compare(password, results[0].password)) ) {
        res.status(401).render('login', {
          message: 'Email or Password is incorrect'
        })
      } else {
        const id = results[0].id;

        const token = jwt.sign({ id:id }, process.env.JWT_SECRET, {
          expiresIn: process.env.JWT_EXPIRES_IN
        });

        console.log("The token is: " + token);

        const cookieOptions = {
          expires: new Date(
            Date.now() + process.env.JWT_COOKIE_EXPIRES * 24 * 60 * 60 * 1000
          ),
          httpOnly: true
        }

        res.cookie('jwt', token, cookieOptions );
        res.status(200).redirect("/");
      }

    })

  } catch (error) {
    console.log(error);
  }
}

exports.register = (req, res) => {
  console.log(req.body);

  const { name, email, password, passwordconfirm } = req.body;

  db.query('SELECT email FROM users WHERE email = ?', [email], async (error, results) => {
    if(error) {
      console.log(error);
    }

    if( results.length > 0 ) {
      return res.render('register.hbs', {
        message: 'That email is already in use'
      })
    } else if( password !== passwordconfirm ) {
      return res.render('register.hbs', {
        message: 'Passwords do not match'
      });
```

```
    }

    let hashedPassword = await bcrypt.hash(password, 8);
    console.log(hashedPassword);

    db.query('INSERT INTO users SET ?', {name: name, email: email, password: hashedPassword
}, (error, results) => {
      if(error) {
        console.log(error);
      } else {
        console.log(results);
        return res.render('register.hbs', {
          message: 'User registered'
        });
      }
    })


  });

}

exports.isLoggedIn = async (req, res, next) => {

  if( req.cookies.jwt) {
    try {

      const decoded = await promisify(jwt.verify)(req.cookies.jwt,
      process.env.JWT_SECRET
      );

      console.log(decoded);


      db.query('SELECT * FROM users WHERE id = ?', [decoded.id], (error, result) => {
        console.log(result);

        if(!result) {
          return next();
        }

        req.user = result[0];
        req.id = req.user.id;
        console.log(req.id);
        console.log("user is")
        console.log(req.user);
        return next();

      });
    } catch (error) {
      console.log(error);
```

```
      return next();
    }
  } else {
    next();
  }
}

exports.logout = async (req, res) => {
  res.cookie('jwt', 'logout', {
    expires: new Date(Date.now() + 2*1000),
    httpOnly: true
  });

  res.status(200).redirect('/');
}

exports.register_admin = (req, res) => {
  console.log(req.body);

  const { name, email, password, passwordconfirm } = req.body;

  db.query('SELECT email FROM admin WHERE email = ?', [email], async (error, results) => {
    if(error) {
      console.log(error);
    }

    if( results.length > 0 ) {
      return res.render('register_admin.hbs', {
        message: 'That email is already in use'
      })
    } else if( password !== passwordconfirm ) {
      return res.render('register_admmin.hbs', {
        message: 'Passwords do not match'
      });
    }

    let hashedPassword = await bcrypt.hash(password, 8);
    console.log(hashedPassword);

    db.query('INSERT INTO admin SET ?', {name: name, email: email, password:
hashedPassword }, (error, results) => {
      if(error) {
        console.log(error);
      } else {
        console.log(results);
        return res.render('register_admin.hbs', {
          message: 'admin registered'
        });
      }
    })
```

```
  });

}

exports.login_admin = async (req, res) => {
  try {
    const {email,password } = req.body;

    if( !email || !password ) {
      return res.status(400).render('login', {
        message: 'Please provide an email and password'
      })
    }

    db.query('SELECT * FROM admin WHERE email = ?', [email], async (error, results) => {
      console.log(results);
      if( !results || !(await bcrypt.compare(password, results[0].password)) ) {
        res.status(401).render('login', {
          message: 'Email or Password is incorrect'
        })
      } else {
        const admin_id = results[0].admin_id;

        const token = jwt.sign({ id:admin_id }, process.env.JWT_SECRET, {
          expiresIn: process.env.JWT_EXPIRES_IN
        });

        console.log("The token is: " + token);

        const cookieOptions = {
          expires: new Date(
            Date.now() + process.env.JWT_COOKIE_EXPIRES * 24 * 60 * 60 * 1000
          ),
          httpOnly: true
        }

        res.cookie('jwt', token, cookieOptions );
        res.status(200).redirect("/admin_page");
      }

    })

  } catch (error) {
    console.log(error);
  }
}
```

Routers (routes/page.js):

```
const express = require("express")
const authController = require('../controllers/auth');
const router = express.Router();
const mysql = require("mysql");

router.get('/', authController.isLoggedIn, (req, res) => {
   res.render('index.hbs', {
     user: req.user
   });
  });

router.get('/register',(req,res) => {
   res.render('register.hbs');
});

router.get('/login',(req,res) => {
   res.render('login.hbs');
});
router.get('/register_admin',(req,res) => {
   res.render('register_admin.hbs');
});
router.get('/admin',(req,res) => {
   res.render('admin.hbs');
});
router.get('/admin_page',(req,res) => {
   res.render('admin_page.ejs');
});
router.get('/profile', authController.isLoggedIn,(req,res) =>{
   if(req.user){
      res.render('profile.hbs',{
         user:req.user,
         id:req.id
      });

   }else{
      res.redirect('/login');
   }

  })

router.get('/create1',(req,res) => {
   res.render('create1.ejs');
});

router.post('/create1',(req,res) => {
   const company_name = req.body.company_name;
   const mkt_cap = req.body.mkt_cap;
   const md_ceo = req.body.md_ceo;
   const bid_price = req.body.bid_price;
```

```
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
         console.log("MYSQL connected.....");
        var query = 'INSERT INTO company(company_name,mkt_cap,md_ceo,bid_price)
VALUES ?';
        var values = [
            [company_name,mkt_cap,md_ceo,bid_price]
        ];

        db.query(query,[values],(err,result) => {
                console.log("company registered");
                db.query("SELECT * FROM company",(err,result)=>{
                    res.redirect("/admin_page");

                });

        });

    }
    })
})

router.get('/admin',(req,res) => {
    res.render('admin.ejs');
});

router.post('/admin',(req,res) => {
    const admin_name = req.body.admin_name;
    const admin_pass = req.body.admin_pass;

    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
         console.log("MYSQL connected.....");
```

```
        var query = 'SELECT * FROM admin_login WHERE admin_name = ? AND admin_pass =
?';
        var values = [
           [admin_name,admin_pass]
        ];

        db.query(query,[values],(err,result) => {
              console.log("Admin verified");
              res.render('create1.ejs');
        });
        }
    })
})

router.get('/company',(req,res) => {
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT * FROM company",(err,result)=>{
            if(err){
                console.log(err);
            }else{
                res.render("company.ejs",{ result: result });

            }
        })}

    })
});

router.post('/company',(req,res) => {
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT * FROM company",(err,result)=>{
```

```
            if(err){
                console.log(err);
            }else{
                res.render("company.ejs",{ result: result });

            }
        })}

    })

});

router.get('/exchange',(req,res) => {
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT * FROM currency_exchange",(err,result)=>{
            if(err){
                console.log(err);
            }else{
                res.render("exchange.ejs",{ result: result });

            }
        })}

    })
});

router.post('/exchange',(req,res) => {
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT * FROM currency_exchange",(err,result)=>{
            if(err){
                console.log(err);
```

```
        }else{
            res.render("exchange.ejs",{ result: result });

        }
    })}

});

router.post('/users',(req,res) => {
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT id,name,email,date_time FROM users",(err,result)=>{
            if(err){
                console.log(err);
            }else{
                res.render("users.ejs",{ result: result });

            }
        })}

    })

});

router.get('/users',(req,res) => {
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT id,name,email,date_time FROM users",(err,result)=>{
            if(err){
                console.log(err);
            }else{
```

```
            res.render("users.ejs",{ result: result });

        }
    })}

  })

});

router.post('/delete',function(req,res){

  const db = mysql.createConnection({
    host: process.env.DATABASE_HOST,
    user : process.env.DATABASE_USER,
    password:process.env.DATABASE_PASSWORD,
    database :process.env.DATABASE
  })
  db.connect((error) => {
    if(error){
    console.log(error);
    }
    else{

      const query = "DELETE FROM company WHERE c_id = ?";
      const c_id = req.body.c_id;

      db.query(query,c_id,(err,result)=>{
        if(err){
          console.log(err);
        }else{

          res.redirect("/company");
        }}
          )}}
)})

router.post('/edit',function(req,res){
  const c_id = req.body.c_id;
  res.render('edit.ejs',{c_id:c_id});
});

router.post("/edit_company",function(req,res){
  const c_id = req.body.c_id;
  const company_name = req.body.company_name;
  const mkt_cap = req.body.mkt_cap;
  const md_ceo = req.body.md_ceo;
  const bid_price = req.body.bid_price;
  const db = mysql.createConnection({
    host: process.env.DATABASE_HOST,
    user : process.env.DATABASE_USER,
    password:process.env.DATABASE_PASSWORD,
```

```
    database :process.env.DATABASE
  })

  db.connect((error) => {
    if(error){
    console.log(error);
    }
    else{

      const query = "UPDATE company SET company_name='"+company_name+"',
mkt_cap='"+mkt_cap+"',md_ceo='"+md_ceo+"',bid_price='"+bid_price+"' WHERE
c_id='"+c_id+"'";


      db.query(query,(err,result)=>{
        if(err){
          console.log(err);
        }else{

          res.redirect("/company");
        }}
          )}}
)
});
router.post('/edit_user',function(req,res){
  const id = req.body.id;
  res.render('edit_user.ejs',{id:id});
});
router.post("/edit_user_submit",function(req,res){
  const id = req.body.id;
  const name = req.body.name;
  const email = req.body.email;
  const db = mysql.createConnection({
    host: process.env.DATABASE_HOST,
    user : process.env.DATABASE_USER,
    password:process.env.DATABASE_PASSWORD,
    database :process.env.DATABASE
  })

  db.connect((error) => {
    if(error){
    console.log(error);
    }
    else{

      const query = "UPDATE users SET name='"+name+"', email='"+email+"' WHERE
id='"+id+"'";
      db.query(query,(err,result)=>{
        if(err){
          console.log(err);
        }else{
```

```
                res.redirect("/users");
            }}
              )}}
)
});
router.post('/add_company',(req,res) => {
    const id = req.body.id;
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    console.log(id);
    db.connect((error) => {
        if(error){
        console.log(error);
        }
        else{
            db.query("SELECT * FROM company",(err,result)=>{
            if(err){
                console.log(err);
            }else{
                res.render("add_company.ejs",{ result: result ,id:id})
            }
        })}

    })

});

router.post('/buy_shares',(req,res) => {
    const id = req.body.id;
    const c_id = req.body.c_id;
    const bid_price = req.body.bid_price;
    res.render('add_form.ejs',{id:id,c_id:c_id,bid_price:bid_price});
});
router.get('/shares_bought',(err,res)=>{

    res.render('shares_bought.ejs');
});
router.post('/final_buy',function(req,res){
    const c_id = req.body.c_id;
    const company_name = req.body.company_name;
    const id = req.body.id;
    const shares = req.body.shares;
    const bid_price = req.body.bid_price;
    const cur_price = req.body.cur_price;

    const db = mysql.createConnection({
```

```
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
   db.connect((error)=>{
    if(error){
        console.log(error);
        }
        else{
           console.log("final id",id);
           var query = 'INSERT INTO holdings(id,c_id,company_name,shares,buy_price) VALUES
?';
           var values = [
           [id,c_id,company_name,shares,bid_price]
        ];



           db.query(query,[values],(err,result)=>{
              if(err){
                 console.log(err);
              }else{
                  console.log("shares bought");
                  console.log(result);
                 db.query("",(err,result)=>{
                 res.redirect("/shares_bought");
              })
              }}
                 )}}
)
}

);

router.post("/holdings",function(req,res){
    const c_id = req.body.c_id;
    const company_name = req.body.company_name;
    const id = req.body.id;
    const name = req.body.name;
    const shares = req.body.shares;
    const bid_price = req.body.bid_price;
    const db = mysql.createConnection({
        host: process.env.DATABASE_HOST,
        user : process.env.DATABASE_USER,
        password:process.env.DATABASE_PASSWORD,
        database :process.env.DATABASE
    })
    db.connect((error)=>{
        if(error){
            console.log(error);
```

```
        }
        else{
            console.log("holdings connected");
            db.query("SELECT DISTINCT(h.company_name),shares,date,buy_price  FROM
holdings h,company c WHERE id = ?",id,(err,result)=>{
                if(err){
                    console.log(err);
                }else{

                    res.render("holdings.ejs",{ result: result,id:id,bid_price:bid_price });

                }
            })
        }
    });
});

router.get('/search',function(req,res){
        const company_name = req.query.company_name;
        const db = mysql.createConnection({
            host: process.env.DATABASE_HOST,
            user : process.env.DATABASE_USER,
            password:process.env.DATABASE_PASSWORD,
            database :process.env.DATABASE
        })
        db.connect((error)=>{
            if(error){
                console.log(error);
            }
            else{
                console.log(company_name);
                db.query("SELECT * FROM company WHERE company_name LIKE
'%"+company_name+"%'",(err,result)=>{
                    if(err){
                        console.log(err);
                    }else{
                        res.render("search_company.ejs",{result:result});
                    }
                });
            }
        });
});
router.get('/back',(req,res)=>{
    res.redirect("/company");
});
module.exports = router;
```

Backend Database:

4.3 User Table

	id	name	email	password	date_time
Edit Copy Delete	1	Mehul Singh	mehul44.ms@gmail.com	$2a$08$PA/tGo.aeEWi3lOtLL7kKOZ1hodmjgAQfEnJtlgUWoY...	2023-01-09 19:28:22
Edit Copy Delete	2	rehul	rahul45@gmail.com	$2a$08$DNsvJq.Z4vWLgVQPGseGd.svC7IPBR/cN49e9wtudXb...	2023-01-09 19:28:22
Edit Copy Delete	3	stock_admin	stocks_manage.45@gmail.com	$2a$08$7hZwS8YIOOfXYe1R8Yuqr.DL5H6VQUfTr1qyBlOyYxC...	2023-01-09 19:28:22
Edit Copy Delete	4	Neeraj kumar	neeraj123@gmail.com	$2a$08$7wz6UdBUkkl/kD9MNxCD9OfcR75Nafz8uc3LgOMGRJk...	2023-01-14 21:54:01

Check all With selected: Edit Copy Delete Export

4.4 Admin Table

	name	email	password	admin_id
Edit Copy Delete	stock_admin	stocks_manage.45@gmail.com	$2a$08$qUiQICSxfoBEaIO9KHHBW.DGOo0vWmI9aRUx5ZIG2tM...	1

Check all With selected: Edit Copy Delete Export

4.5 Company Table

	c_id	company_name	mkt_cap	md_ceo	bid_price
Edit Copy Delete	1007	Tata Steel Limited	141486	Thachat Narendran	115.35
Edit Copy Delete	1009	Adani Transmission Limited	288394	Anil Sardana	2605.2
Edit Copy Delete	1016	Titan Company	225329	Coimbatore venkataraman	2484.65

Check all With selected: Edit Copy Delete Export

4.6 Holdings Table

	id	c_id	company_name	shares	date	buy_price
Edit Copy Delete	1	1009	Adani Transmission Limited	69	2023-01-12 14:39:09	2463.2
Edit Copy Delete	1	1016	Titan Company	34	2023-01-12 14:42:10	2484.65
Edit Copy Delete	1	1016	Titan Company	2	2023-01-12 14:54:33	2484.65
Edit Copy Delete	1	1009	Adani Transmission Limited	100	2023-01-12 15:09:07	2605.2
Edit Copy Delete	4	1009	Adani Transmission Limited	12	2023-01-14 22:08:29	2605.2
Edit Copy Delete	4	1016	Titan Company	10	2023-01-16 18:53:28	2484.65
Edit Copy Delete	4	1007	Tata Steel Limited	50	2023-01-16 18:53:51	115.35
Edit Copy Delete	4	1009	Tata Steel Limited	2	2023-01-16 18:54:56	2605.2
Edit Copy Delete	4	1016	Titan Company	3	2023-01-16 19:01:15	2484.65

Check all With selected: Edit Copy Delete Export

4.7 Sold Table

	id	company_name	shares	date_time
Edit Copy Delete	1	Adani Transmission Limited	69	2023-01-17 10:16:52
Edit Copy Delete	1	Adani Transmission Limited	100	2023-01-17 11:44:55
Edit Copy Delete	1	Titan Company	2	2023-01-17 11:33:44
Edit Copy Delete	1	Titan Company	34	2023-01-17 10:21:04
Edit Copy Delete	4	Titan Company	10	2023-01-17 12:54:51
Edit Copy Delete	5	Tata Steel Limited	52	2023-01-17 15:04:31

4.8 Currency_Exchange Table

	currency_name	exchange_rate
Edit Copy Delete	Australian dollar	0.01773
Edit Copy Delete	British pound	0.010108
Edit Copy Delete	canadian dollar	0.016471
Edit Copy Delete	Euro	0.011377
Edit Copy Delete	Japanese Yen	1.59759
Edit Copy Delete	US Dollar	0.012314

Check all *With selected:* Edit Copy Delete Export

4.9 Support Table

	token_number	id	concern	date_time
Edit Copy Delete	145680	4	kiop	2023-01-15 18:03:55
Edit Copy Delete	145682	1	please update my email	2023-01-16 21:34:41

Check all *With selected:* Edit Copy Delete Export

CHAPTER 5

INTERPRETATION OF RESULT

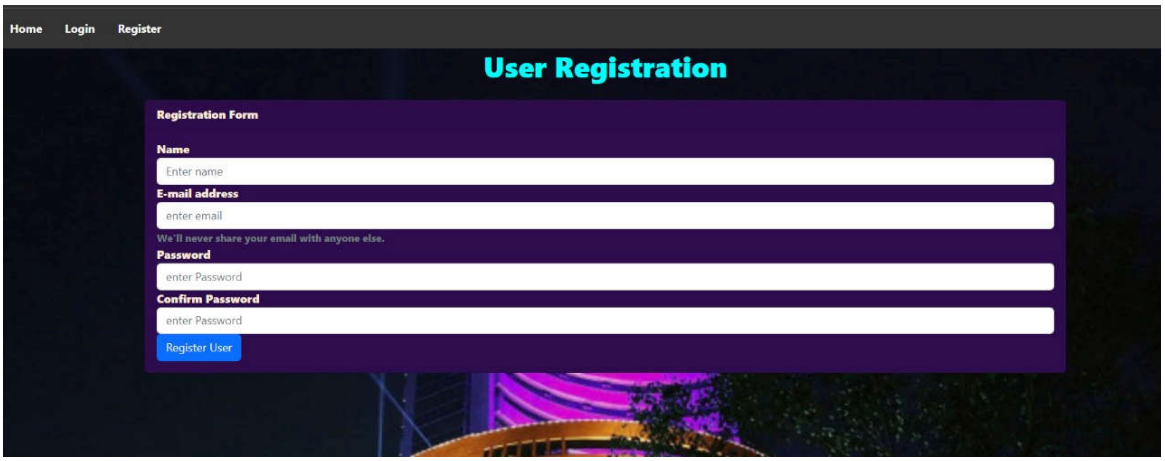

FIG 5.1 USER REGISTRATION PAGE

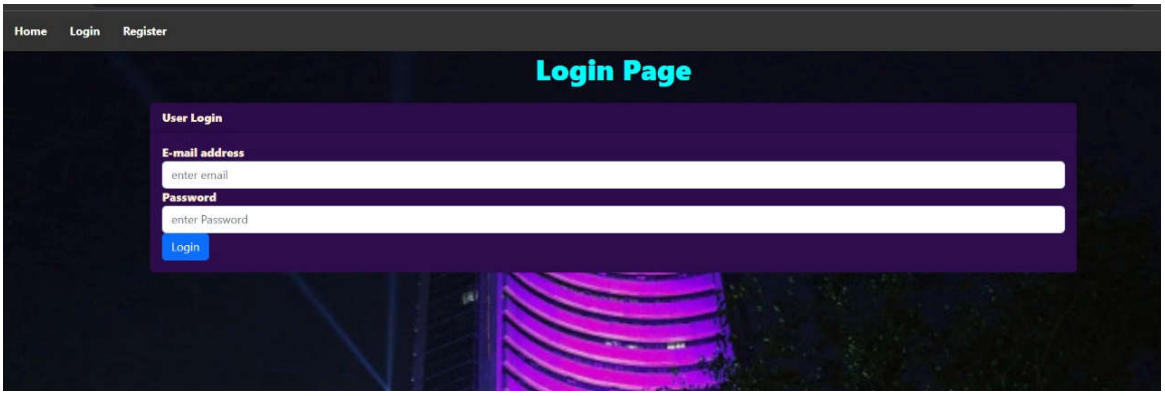

FIG 5.2 USER LOGIN PAGE

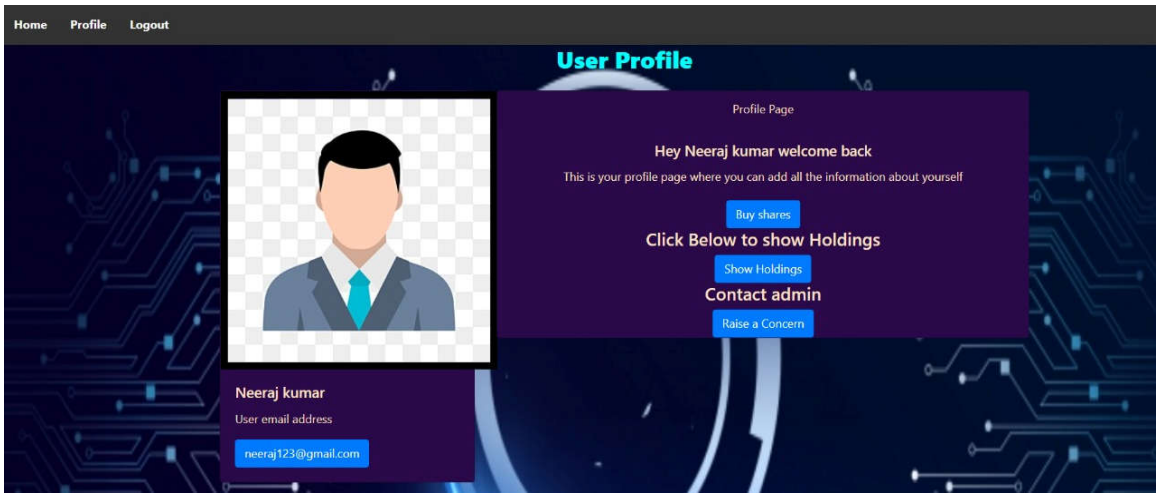

FIG 5.3 USER'S PROFILE

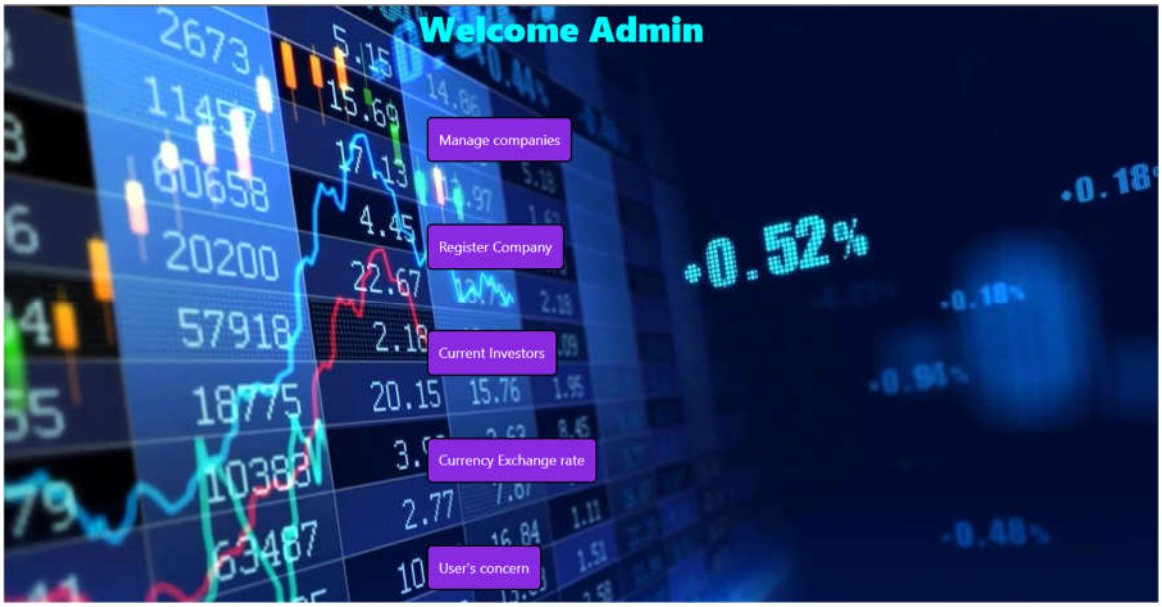

FIG 5.4 ADMIN PAGE

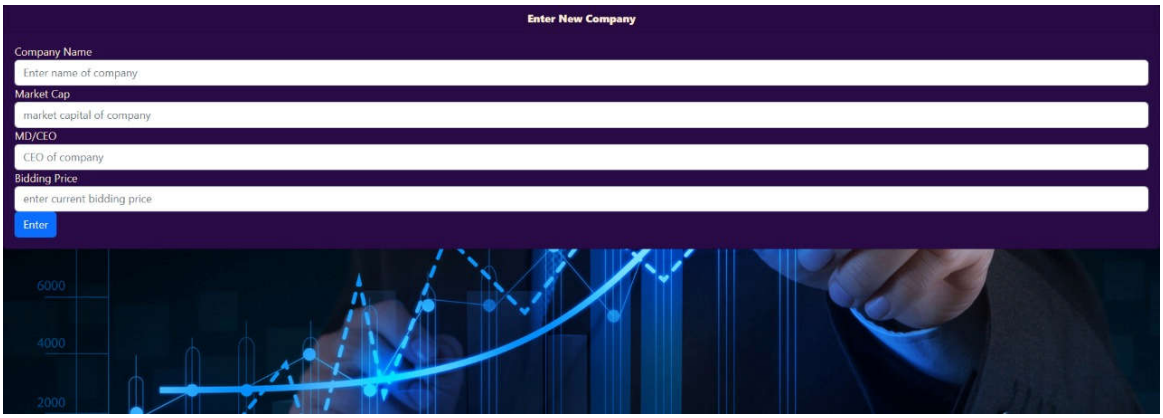

FIG 5.5 NEW COMPANY REGISTRATION PAGE

FIG 5.6 COMPANIES LIST PAGE

FIG 5.7 CURRENT REGISTERED USERS PAGE

FIG 5.8 USERS'S HOLDINGS PAGE

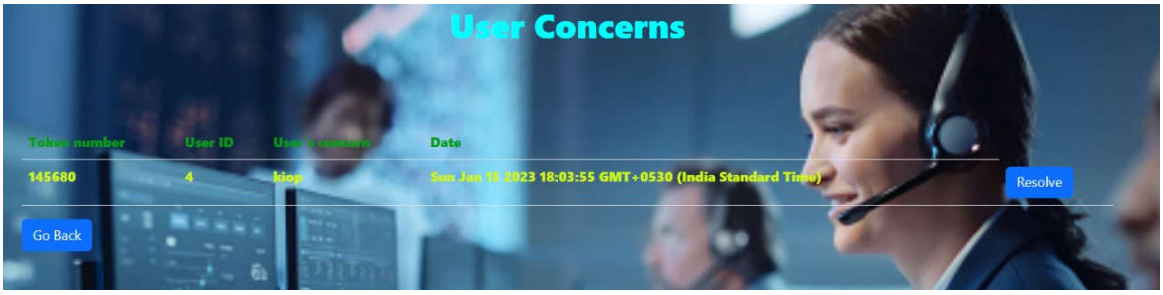

FIG 5.9 USER'S CONCERNS PAGE

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

Although our ambition was to achieve a complete system that will have a highly accuracy. We have managed to develop a system and a guideline to, on how an application can be developed for Stock Market Management System based on web. We have face with many errors and fixed it. We designed this application interface which is simple and well organized so that the user can easily cooperate with it. We tried to improve its accuracy and keep it acceptable by user. We tried our best for best result and keep user friendly And We hope the user will be benefitted.

We learned how to architect a web application. Doing this project we got to know about many new things and we did a proper use of internet. And learned how to complete the work according to the plan. There has no major limitation in this project. We finished the project as supervised but if there are any other user requirements then this will be developed according to the requirements.

REFERENCES

[1] NodeJS:

URL: https://nodejs.org/docs/latest-v16.x/api/

[2] ExpessJS:

URL: https://expressjs.com/en/guide/routing.html

[3] YouTube:

URL: https://youtube.com/watch?v=EN6Dx22cPRI&feature=shares

[4] Bootstrap:

URL: https://getbootstrap.com/docs/4.0/components/forms/

[5] Textbook:

Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson.